A FPGA and Zernike Moments Based Near-field Laser Imaging Detector Multi-scale Real-time Target Recognition Algorithm

LIU Li-feng, MA Hui-min*, LU Ming-quan
Department of Electronic Engineering
Tsinghua University
Beijing, China
llf08@mails.tsinghua.edu.cn, mahm@ee.tsinghua.edu.cn, lumq@mail.tsinghua.edu.cn

***Abstract*—Laser imaging Detector is one direction of Detectors' development. The encounter attitude of the detector and aircraft can be arbitrary, and there may be clouds and smoke– thereby causing interference to imaging. This paper presents a stable algorithm based on Zernike moments which can effectively recognize flying plane and against the cloud and smoke interference. The paper attempts to calculate Zernike moments of an image by using FPGA, and treat them as some kinds of feature to identify target by using the Fisher classifier which can achieve a good classification result.**

Keywords-FPGA, Zernike moments, multi-scale, near-field laser imaging Detector, real-time, target identification

I. INTRODUCTION

Laser imaging Detector can get an image with the following characteristics[1]:

- Full space imaging. The detector and aircraft may be in any posture when encounter happens, so the detector may acquire a projection of aircraft at any point of view.
- Strong noise. Because of cloud and smoke interference, imaging process is often accompanied by strong noise which may lead to error action of detector – one of the most important reasons of an accident.

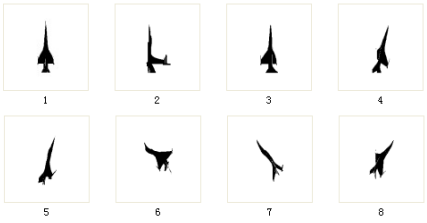

Figure 1. Some examples of imge.[1]

Here are some examples of images, which are generated by three-dimensional simulation software[2], as shown in Fig. 1.

So the requested algorithm for identifying target must have the following characteristics[2]:

- High-speed. If the detector has flew over an aircraft far away when the algorithm gives the recognition result, there is no meaning already. This requires the algorithm must be fast enough, so the detector will give the recognition result before it flies over the aircraft too far away.
- Robust. This character makes the algorithm resist the strong noise.
- Realizable. The algorithm must be able to be implemented in the present hardware. If this is not met, then no matter how fast, or how robust the algorithm is, it can only be "on paper." This requires the algorithm must be simple enough.

This paper attempts to extract the Zernike moments of a detector image and treat them as one kind of feature to identify the target.

Obviously, Zernike moments are not simple features, how to implement them in hardware and design an classifier which can meet the requirements for identifying the targets fast and correctly, is the major work of this article. Firstly, this paper will briefly introduce previous work and Zernike moments, and then, will give the process of its implement in hardware(FPGA), and finally, will give the recognition process, and the recognition results to make a discussion.

II. PREVIOUS WORK

[2] designs and develops a near-field laser imaging detector target identification system, which provides a hardware support for our algorithm. Our algorithm is implemented in its hardware constraints. [2] also constructs a near-field laser imaging detector simulation system, through which we can easily obtain the target images. This provides test data for our algorithm, and a basis for comparison.

[1] Supported by National Natural Science Foundation of China (60502013) and the National High Technology Research and Development Program(863 Project) of China(2006AA01Z115).
*Corresponding author.

978-0-7695-4360-4/10 $26.00 © 2010 IEEE
DOI 10.1109/ISISE.2010.78

[3] studies some target recognition methods of laser imaging detector in theory, but do not give the process of hardware implementation and the recognition results.

[4] and [5] present a method to obtain the skeletons of an image, and use them as a feature to identify the aircraft. This feature can be implemented in the hardware to meet the real-time requirements, the recognition rate reaches 95%.

[6] gives four target recognition algorithm, namely, the accumulation of pixels, connected domain detection, abrupt change of image edge and statistics center line. The recognition rate is above 95% by using the four kinds of algorithms comprehensively. Features they use are relatively simple and therefore can be easily implemented in hardware to achieve real-time target recognition.

The biggest problem of [4][5][6] is that they do not consider the cloud and smoke interference to laser imaging detector, and don't test the algorithm with cloud images, so they have not given the error rate. The recognition rate is not reliable without the error rate.

The result in this paper will show that our algorithm can not only get a higher recognition rate but also get a lower error rate which is obtained by testing the algorithm with cloud images generated by simulation or actually captured in laboratory.

III. ABOUT ZERNIKE MOMENTS

Zernike moments were proposed by the Zernike. They are defined as follows[7]:

$$Z_{\mathrm{nm}} = \frac{n+1}{\pi} \sum_i \sum_j f(x_i, y_j) V_{\mathrm{nm}}^*(x_i, y_j), \quad (1)$$

where n is positive integer or zero, and m is positive or negative integers subject to constraints $n-|m|$ even, $|m| \le n$ (Because $R_{\mathrm{nm}}(\rho) = R_{\mathrm{n,-m}}(\rho)$, let the m>0 in our paper), and V_{nm} is defined as:

$$V_{\mathrm{nm}}(x_i, y_j) = V_{\mathrm{nm}}(\rho\cos\theta, \rho\sin\theta) = R_{\mathrm{nm}}(\rho)\exp(jm\theta), \quad (2)$$

where the $R_{\mathrm{nm}}(\rho)$ is defined as:

$$R_{\mathrm{nm}}(\rho) = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!}{s!(\frac{n+|m|}{2}-s)!(\frac{n-|m|}{2}-s)!} \rho^{n-2s} \quad (3)$$

.

ρ is the polar axis of an image pixel in the normalized polar coordinates, while θ is the corresponding polar angle. $f(x_i, y_j)$ is the pixel gray value corresponding to the location (x_i, y_j), which is 0 or 1 for binary images.

Zernike moments were used for image reconstruction and recognition[7] (conducted by Alireza Khotanzad and YAW HUA HONG, Southern Methodist University), and they had achieved good results.

Zernike moments have some good characteristics, such as translation, rotation, and scale invariance[7].

Relative to the point, edge features, Zernike moments are surface features, so they are more stable characteristics.

Note that for a fixed order of the Zernike moments, the order of n, m and factorial could be fixed, so the order of the coefficients of ρ^{n-2s} could also be fixed. This will greatly facilitate the calculation of Zernike moments, which will make it possible for the hardware to calculate Zernike moments.

The following table gives an example:

TABLE I. EXAMPLE OF N,M AND THE COEFFICIENTS

N	0	1	2			3			4		
M	0	1	0		2	1		3	0		
Coefficient of ρ^{n-2s}	1	1	-1	2	1	-2	3	1	1	-6	6

IV. COMPLEX NUMBER DESIGN

Ideally, Zernike moments should use floating point number for calculations. But in FPGA, floating-point calculation implementations are very time-consuming and resource intensive.

To overcome this difficulty, we propose a method – to achieve high precision arithmetic with fixed-point decimal. Fixed-point decimal format is as shown in Fig. 2.

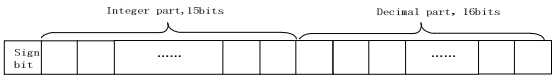

Figure 2. Fixed-point decimal format.

However, note that not all arithmetic of Zernike Moments needs to use decimal – the decimal format above is so large, in order to save valuable register resources in the FPGA, so, a hybrid number system is actually used in this paper. Some variables, such as counters and variables n, m of Zernike moments, is represented by integer, and some other variables, such as the values of sum, is represented by complement code. We introduce the fixed points to facilitate the calculation, especially in the operations of multiplication and division. The complement code is mainly used for addition, subtraction operations.

V. ARITHMETIC IMPLEMENTATION

Observe the composition of Zernike moments in (1), in order to get the Zernike values of an image, one needs to use addition, subtraction, factorial, power, trigonometric functions, complex arithmetic operations and so on. As can be seen, almost all of the basic operations in mathematics are used. After using the complex number system described above, the floating-point arithmetic units can be replaced by fixed-point arithmetic units which are equivalent to integer arithmetic units.

The implementation of various arithmetic operations in this paper are as follows:

- The decimal multiplication, division, and extraction of square root units can be replaced by fixed-point multiplication, division and rooting units, easily using modules generated by Quartus II's MegaWizard tools.

- Because the order of using factorial operations is fixed as in table I, their values can be computed in advance, you just need to look-up a table when using.
- Trigonometric function values can be obtained by looking-up tables.
- Complex can be separated to the real and imaginary components.
- Complex exponentiation can be expanded through using Euler's formula.

Power operation is more difficult to achieve. In this paper, only four registers are needed to calculate all the values of power operations easily by changing the order of computing.

Let:

$$c_s = (-1)^s \frac{(n-s)!}{s!(\frac{n+|m|}{2}-s)!(\frac{n-|m|}{2}-s)!}, \quad (4)$$

then for fixed n and m:

$$R_{nm}(\rho) = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!}{s!(\frac{n+|m|}{2}-s)!(\frac{n-|m|}{2}-s)!} \rho^{n-2s} = \sum_{s=0}^{(n-|m|)/2} c_s \rho^{n-2s}. \quad (5)$$

Let:

$$k = n - 2s, \quad (6)$$

then:

$$s = (n-k)/2. \quad (7)$$

$k = n$ when $s = 0$; and $k = |m|$ when $s = (n-|m|)/2$.

So:

$$R_{nm}(\rho) = \sum_{k=|m|}^{n} c_k \rho^k \quad (8)$$

Because $|m| \le n$ is always correct, ρ^{k_1} is always calculated before ρ^{k_2} ($k_1 \le k_2$).

TABLE II. ANALYSIS OF POWER ARITHMETIC

	Do not use values calculated before		Use values calculated before	
Exponent	Exponent arithmetic	Multiplicati-on times at least	Value is saved or not?	Multiplicati-on times
0	0	0		0
1	1	0		0
2	1+1	1	Saved	1
3	2+1	2	Saved	1
4	2+2	2	Saved	1
5	3+2	3	Saved	1
6	3+3	3		1
7	4+3	4		1
8	4+4	4		1
9	5+4	4		1
10	5+5	4		1

After this transformation, a lot of resources and time can be saved. When the maximum n is 10, analysis as table II above.

VI. PIPELINE DESIGN

Division is used at several parts of the algorithm. These division operations do not run at the same time, so just one divider is enough to be used in the pipeline as a result of which a lot of FPGA resources are saved.

Because a lot of looking-up table operations are used when arithmetic operations are implemented, if all the values are stored by the internal registers of FPGA, there is no doubt it will be the fastest and most efficient. All of the sine, cosine values can be instantly obtained. However, it is unfortunate that, FPGA do not have enough internal registers. The FPGA being used is EP2S30F484C5N of Altera corporation, which has only 143520 registers[8], so we can only use a lot of RAMs and ROMs. There are so many RAMs or ROMs in the design, such as 'acos ROM', 'cos ROM', 'sin ROM', 'sita RAM', 'R RAM', 'input image RAM' and so on. The values got from RAMs or ROMs could really be used every other cycle when the clock of the RAMs or ROMs is the opposite of main clock. This involves a pipeline issue, otherwise, the time for calculating Zernike moments will be doubled.

The most complex calculations are the ones involving angles. First, the angle corresponding to a pixel of any image in the normalized polar coordinates should be taken out from the RAM first. At the same time, the angle of its m-fold is calculated, and then the sine, cosine value of the m-fold angle are calculated, which should be taken out from another two ROMs, and finally the values of trigonometric functions should multiply with the corresponding polar axis of the pixel which is taken out from the other piece of RAM. All above makes the timing relationships of this calculation very complicated .

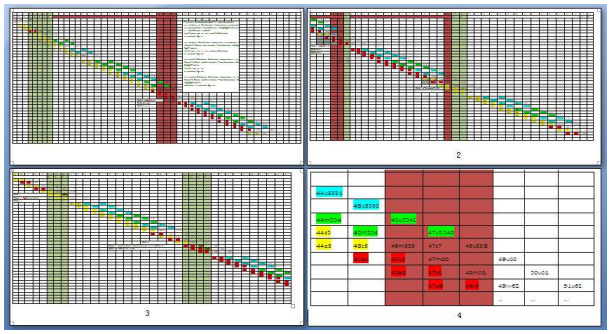

Figure 3. Analysis on the pipeline about angle.

Analysis on the pipeline about angle is shown in Fig. 3. Each column in the table represents a clock cycle, and the columns identified with a light green represent that the corresponding image pixel value is 0. So the subgraphs of 1,2,3 are corresponding to the situation that the number of pixels valued 0 between two pixels valued 1 is 5,0,1,2,3,4 in turn. When the pixel value is 0, the Zernike moments of that pixel is 0; there is no need of calculation, so only one cycle is needed–this can save a lot of time. If the interval between the two pixels valued 1 is greater than or equal 6, the pipeline

situation is the same as that of 5–that is because at least 5 cycles are needed from the address of 'sita RAM' is valid, to the corresponding angle value is taken out the 'sita RAM', to the addresses of the 'cos RAM', 'sin RAM' change to be valid, and finally, to the values of the trigonometric functions of m times the angle are obtained from the ROMs. Analysis of subgraph 4 is the case two pixels valued 1 are adjacent, the timing for the pipeline of which is the most critical.

Fig. 4 is a magnification of subgraph 4 in Fig. 3. The figures before the letters represent the number of cycles. The letter 'a' represents that the address of 'sita RAM' is valid, followed by a number representing pixel positions; the letter 's' represents that an angle has been obtained at the negedge of a clock, and the number after 's' also represents the location of a pixel; the letter 'm' represents that sita got from 'sita RAM' can be used at the rising edge of next cycle, and m-fold angle can be obtained by multiplying m with sita, the number after m represents the location and order. Once the m-fold of angle sita is calculated, it means the addresses of cosine, sine ROM are valid, then the values can be used a cycle after with the rising edge of clock – 'u' is on behalf of 'usable', followed by the number representing the location, order, and a counter k. As can be seen from the chart, for the pixel 6 (start counting from 0), to get the sine and cosine values it first uses, there is a need to enable the corresponding address of 'sita RAM' when calculating order 33 Zernike moments with counter k is 0 of pixel 5. For other cases, there is no need to make the address of 'sita RAM' valid at this moment, but also a similar analysis.

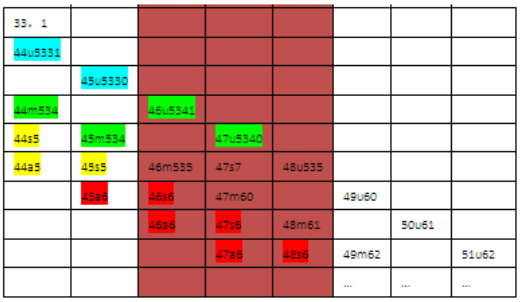

Figure 4. A magnification of subgraph 4 in Fig. 3.

VII. PSEUDO-IMAGE SEGMENTATION

For our algorithm, we certainly hope that image segmentation has already been done before making Zernike moments feature extraction, but in the hardware environment, the image segmentation is very difficult to implement. Because Zernike moments calculation has occupied a lot of the FPGA resources, we simplify the image segmentation, that is "pseudo-segmentation". The reason why called "pseudo-image segmentation" is that this segmentation is so simple. Simple as it is, but for the calculation of Zernike moments, and, ultimately, target recognition, it is indeed essential.

Because every image is obtained line by line during the imaging process of detector, we can get that the image which has been received and the image after are two completely different objects when there are N lines continuously having few pixels. By this, a simple image segmentation can be implemented. In our experiment, N is 16.

An example of pseudo-image segmentation is as Fig. 5.

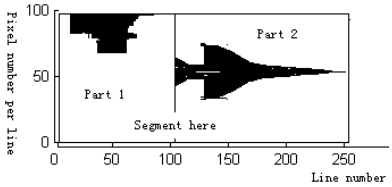

Figure 5. Pseudo-image segmentation.

Total 256 lines are obtained from left to right in Fig.5, and each line has 96 pixels. There are 16 lines having few pixels continuously obtained after part 1 is got, so a segmentation is done. Then the image obtained is part 2.

VIII. MUTI-SCALE OBJECT RECOGNITION

Despite the use of design above, we find that the image is so large that the calculation of a 256*96 size image Zernike moments feature is still very time-consuming. So, in order to calculate Zernike moments more rapidly, we scale the image first. Image scaling has several benefits:

- The calculation of Zernike moments is much more easier.
- There is no need of scaling when only having a few rows of pixel, distinguish image directly, then the image has a higher precision and the recognition can be considered as a details recognition.
- There is a need for image scaling when receiving many rows of data, and then for target recognition. At this time, the details of image are not clear and the identification of the image is closer to a global identification.

The recognition process above is more coincident with the imaging process of detector, and with the recognition process of human.

Details recognition is of a less well recognition method. Because, if the laser imaging detector "sees" a triangle, it is not necessarily one of the aircraft wings, it may also be a bizarre cloud.

The identification process is as Fig. 6:

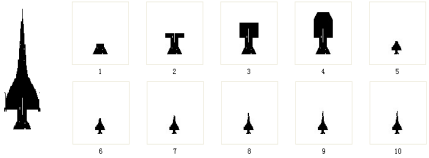

Figure 6. The identification process.

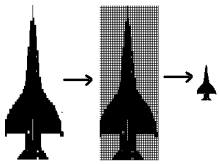

Figure 7. The specific process of scaling.

The scaling process is like this: the 256*96 image is zoomed out to a 64*32 image, which means 1 pixel of the shrunken figure corresponds to the 4*3 pixels of the original image. The specific process is as Fig. 7 above.

We use the Fisher classifier to identify the target, which can distinguish aircrafts from clouds, smoke and so on relatively well.

IX. RESULTS AND DISCUSSION

As described above, we implement the calculation of Zernike moments on a FPGA of Altera corporation: EP2S30F484C5N. At the same time, we implement the image scaling and classifier design. The total use of resources is 87%. That is to say, the algorithm is still very complex, and most resources of FPGA are used.

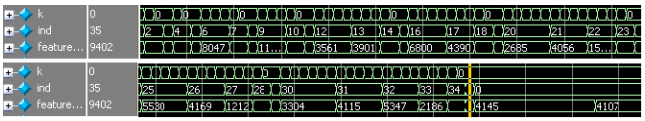

Figure 8. The results of FPGA

For the 64*32 image in Fig.7, compare the results calculated in hardware with that by Matlab, the result of FPGA is as Fig. 8 above. 25540 clock cycles are used.

Table III shows the detailed comparison of results of FPGA and Matlab, and gives the deviation. As the hardware uses the 16 bit decimal, the results of Matlab are multiplied with 65,536 in order to facilitate comparison.

TABLE III. COMPARISON THE RESULTS OF FPGA AND COMPUTER

Order	1	2	3	4	5	6	7	8	9
Matlab	20861	0	43899	6400	8064	3891	45160	11473	4771
FPGA	20776	1	43870	6395	8047	3885	45070	11449	4761
Deviation(%)	0.41	——	0.07	0.08	0.21	0.15	0.2	0.21	0.21
Order	10	11	12	13	14	15	16	17	18
Matlab	11659	3565	3903	30547	14308	6807	4395	8320	5403
FPGA	11645	3561	3901	30515	14290	6800	4390	8310	5396
Deviation(%)	0.12	0.11	0.05	0.1	0.13	0.1	0.11	0.12	0.13
Order	19	20	21	22	23	24	25	26	27
Matlab	2685	4057	15296	9662	9839	5534	4172	1218	9768
FPGA	2685	4056	15291	9654	9833	5530	4169	1212	9748
Deviation(%)	0	0.02	0.03	0.08	0.06	0.07	0.07	0.49	0.2
Order	28	29	30	31	32	33	34	35	36
Matlab	1106	3307	4120	5333	2196	9226	9406	4146	4109
FPGA	1107	3304	4115	5347	2186	9228	9402	4145	4107
Deviation(%)	0.09	0.09	0.12	0.26	0.46	0.02	0.04	0.02	0.05

The result shows the image Zernike moments calculated by FPGA are still very high accuracy.

To illustrate the effectiveness of the algorithm in this article, and in order to compare with each other, we have the same test conditions as [6]. We also add some cloud interference images, some of which are shown in Fig. 9.

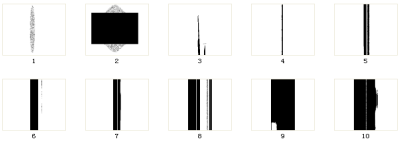

Figure 9. Example of clouds.

Subgraphs 1, 2 are from [9], and the rest are from laboratory test. Test results are as table IV:

TABLE IV. TEST RESULTS

	Recognition rate		Error rate	
	Details recognition	Global recognition	Details recognition	Global recognition
Our method	100%	98.7%	14.0%	3.8%
Yu's method[6]	95%		20.5%	

As can be seen above, our algorithm can get better recognition results under the same conditions. The error rate of details recognition is much higher than that of global recognition, that is because the global identification could obtain a more global information, while details recognition don't. Even so, the error rate is still lower than the algorithm of [6].

ACKNOWLEDGMENT

Thanks my tutor Ma Hui-min for giving my help during the thesis, her profound knowledge and strict spirit let me sincerely admire. Thanks Yu Xiao-liang, I would not understand the laser imaging detector simulation recognition system so quickly without his help. Also thanks Pang Bo, he has a deep understanding of Zernike moments, I have discussed with him many times during the thesis. Finally, thanks Li Juan for inspiration to me.

REFERENCES

[1] WEI Bin, ZHENG Lian, WANG Ke-yong, "A New Method for Laser Fuze Imaging Detection", Opto-Electronic Engineering, 32(2005), 36(in Chinese).

[2] YU Xiao-liang, "Design on Imaging Simulation and Signal Processing System Based on Laser Near-field Detection", TsinghuaUniversity, Beijing, 2009(in Chinese).

[3] WANG Yan, ZHENG Lian, WANG Ke-yong, "Local feature extraction technology used for image recognition of imaging fuze", Infrared Technology, Vol.23, No.5, Sep. 2001(in Chinese).

[4] SONG Cheng-tian, WANG Ke-yong, ZHENG Lian, "The image processing and target identification of laser imaging fuze", Proceedings of 2008 3rd International Conference on Intelligent System and Knowledge Engineering, ISKE 2008, Xiamen, China, 2008: 1117-1120.

[5] SONG Cheng-tian, WANG Ke-yong, ZHENG Lian, "An image processing system research on target and rendezvous status identification of missile and plane", Proceedings - 1st International Congress on Image and Signal Processing, CISP 2008, Sanya, Hainan, China, 2008: 525-527.

[6] YU Xiao-liang, MA Hui-min, XIAO Jian, "Real-time target recognition system simulation based on laser near-field detection", Optoelectronics Letters, Vol.5, No.4, 1 July 2009.

[7] Alireza Khotanzad and YAW HUA HONG, "Invariant Image Recognition by Zernike Moments", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.12, No.5, May 1990.

[8] Stratix II Device Handbook, Altera Corporation, 2007.

[9] SHI Kang, "Air Target Modeling and Real-time Processing Algorithm ", Tsinghua University, Beijing, 2008(in Chinese).